A Formal Framework for SAAS Customization Based on Multi-Layered Architecture via Process Algebra

Zubair Muhmmad *m.zubair_21@yahoo.com*
School of Computer Science and Engineering,
Nanjing University of Science and Technology,
Nanjing, 21000,China.

Chalise Birendra *biren448@gmail.com*
School of Computer Science and Engineering,
Nanjing University of Science and Technology,
Nanjing, 21000,China.

Dongmei Liu *dmliukz@njust.edu.cn*
School of Computer Science and Engineering,
Nanjing University of Science and Technology,
Nanjing, 21000,China.

Abstract

As the rapid increment on the number of software systems and its's user, the complexity to manage the software systems is not very easy. Software as a service (SaaS) provides only user required software services in form of web - mostly based on the vendor developed/maintained model, which creates the new challenges for the software customers (tenants).

In this paper, we purpose a multi-layered architecture of SAAS framework, customized by both vendors and tenants - with the help of process algebra. Moreover, the architecture will be able to offer an extant vendor model of SAAS as well as tenant based precise self-customization services system, while all the processes are present in an algebraic form.

Finally, we show the efficiency and effectiveness of our architecture via process algebra, which we believe is a well-designed and non-existing architecture of the SAAS customization framework.

Keywords: SAAS, Software Customization, Process Algebra, Multi-layred, Architecture.

1. INTRODUCTION

Software as a service (SAAS) as discuss in [1], is a new enterprises model for accessing software services of required vendor via internet networks without installing software. In recent days, the rapid growth of software systems and services changes the enterprise's availability and cost of products massively.

However, there are a lot of issues of efficiency and effectiveness with existing architectures. While providing services to different and large number of tenants, vendor faces the challenge of customization. Here, we are going to introduce a framework for SAAS customization based on multi-layered architecture, which provides a well-designed strategy to enable self-service customization and configuration by their customers without changing the main application source code for any customers.

In this paper, we explain the multi-layered architecture[9] to customizing different type of tenant services as well as all possible processes of customization through the process algebra. The

framework is further modified by vendors as well as tenants through self-customization system and its defining rules, which is concurrently supported by SAAS application. These applications are responsible for translating and managing rules and information. All the information is store in the database, and service is offer to customer according to the rule of SAAS application [7]. Moreover, the customizations is introduce in the form of processes and defines via algebraic rules for all tasks aka processes of both vendor and tenants.

For the first time, this paper describes the design and implementation of both vendor and user customizable SAAS architecture using process algebra [5], which provides a comprehensive evolvement over existing architectures.

2. RELATED WORKS

Vendors and Tenants' customization of SAAS via process algebra is relatively new issue; although a lot of researches are exist on the customization of a traditional vendor models. The achievement of the relatively new problems is always based on some ground rules of existing works.

The configuration and customization competency model [1], provides the concept of potential of the tenant configuration and customization but fails to create the programming models and environment, and its runtime behaviors. Similarly, Template based business logic customization for SaaS applications [2, 3], describes the architecture on the vendor models, business rule templates, and flexible business logic customization but completely ignores the tenant issues.

However, Software Customization Based on Model Driven Architecture over SAAS Platform [4], proposes the templates based customization concepts of service models as existing old fashioned business or vendor based models. And Data Privacy Preserving Mechanism based on Tenant customization for SAAS[8] describes the tenant data privacy policy, which protects the users data from another user or third-party accesses.

While this paper, a formal framework for SAAS customization based on multi-layered architecture via process algebra provides business and customer customizable architecture.

3. DEFINING CUSTOMIZATION RULES

There are different types of customization, and tenant can modify either one or all. They can customize their project structure according to requirements or organization's structure. So here, tenant can customize Organization structure different, Workflow difference, way to process data, Data difference, user interface difference, Different reports, and business rules difference.

Organization structure difference: If tenant organization structure [7] is different than Vendor's. Following three different categories are customizable:

1. Add new roles 2. Change roles or modify 3. delete roles

According to the software requirements vendors add some roles in main database roles table. When a new tenant register into system, s/he selects the default setting - if they want to use all roles added by vendors, or tenant select the customized setting to add role and use his own roles. But if setting is self-customized then tenant cannot see vendor's roles and other tenant roles. So, in database role table there will be only three columns as role id, role text and add.

However, when required users can add new role into the role table and assign staff roles in the staff table or modify the existing roles. Users also can delete the role from role table, only which is added by them but they can't delete role added by other tenant or vendors[8].

Workflow difference: If tenant's organization workflow [12] is different from existing workflow, tenant is allowed to change roles sequence, add new sequence, reorder or temporarily on/off

task. According to software requirements, vendors can add some task, process it and put it into the table. Vendors also can define the process steps and complete it. At the registration time, if tenant selects the default setting it is the vendors setting. But if tenant selects the customization setting, all tasks, processes and the process steps for that tenant is self-customizable.

However, the Tenant can't see the other tenant task and process. He can only see and process his own task and modify it according to his own needs. All tenants only can change the roles sequence, add new task and reorder the tasks of their own. They can't change vendors' added task, other tenant tasks or processes but tenant can temporarily on off tasks.

Way to process data: In this part, vendors can add some action and trigger it, but if tenant think that his data processing way is different from vendors, he can add some action and trigger it like other actions at the registration time. But if select the default setting or all vendors action and trigger, it will be treated as a tenant action and trigger. Otherwise, tenant need to add his own action and trigger the action into trigger table as the following three steps rules according to his requirements and it will not affect the main database.

1. Add new action 2. Trigger action 3. Change action/steps

On satisfaction of the trigger condition, system executes the tenant setting for defined action. So, tenant can performed more than one steps on one trigger execution and change the action steps according to his requirements via setting.

Data difference: When tenant think his data is different, vendors add the data and the customization includes the facility of add new data according to tenants' product to manage. Tenant can add a new data after registration by the name of his attributes into attributes table. So, when he needs to change the data he can select the data from attributes table with his id. Here, tenant performs three changes (Add new data, Change existing data, and Delete data) as if his data is different than vendors' added data. But when tenant need to change existing data then he can perform one of the following steps:

1. Add custom field 2. Change field name 3. Change field type

If tenant needs to add new data he can select the data from his attributes table, which was added at the registration time. If he needs to change the existing data then he can use new field into table or change field name based on already added or tenant's need to change the field type. If tenant has no need of existing data he can also delete the data but only from his own database not from main database or other tenant database[1].

User Interface Difference: In this part, vendors provide different type of user interface at **UI**[10]. Tenant can select the interface according to his choice and perform different steps on his own.

- Reflection or change of data at UI
- Change look and feel (color font layout etc.)
- Present extra information

In extra information section, tenant can add different component and make changes to add new hyperlink and new component (chart, graph etc.).

Here, all the required tables are created in the tenant's database at the time of registration in main database, so vendors can create all the same tables and store as default setting. At first, all default setting updated into tenant database and they can change later.

Different reports: Tenant can select the different reports style and query of rules according to his requirements and vendors can add the defaults setting in the main database. All the tables created at registration time as default value in tenant database, so he can change later. Tenant

can perform the three main changes as: Add or change the reports style, Add or change query rules, and add or change data sets.

Different business rules: In this section, vendors provide the facility to tenant that can make the organization rule and set the different trigger for rules, and tenant can perform the two main functions as Change / set rules and change / set rules trigger.

Rule Translator: When tenant select and complete the different category of customization, it is send to the rule translator. Then, Rule Translator execute predefines vendor's rules or user's actions from the databases in order to manage the tenants' customization.

Rule Manager: The rule manager is responsible for managing customization into services and processes, which is further handled by existing databases. It also controls the access over the databases and its contents to protect from the misuses according to vendor's rules.

Service and Process layers: In SAAS applications, service and process layers shows the each and every customization service and processes. For each unique customization, there might be a multiple process and services according system's rules.

4. ARCHITECTURE OF THE FRAMEWORK FOR SAAS CUSTOMIZATION

In this figure (figure 1.1) we show a well-designed for architecture for SAAS customization. The architecture is based on the pre-defined customization rules (described in section 3).

Here, we are presenting a quite simple working mechanism for multi-layered architecture[9]. The users (tenant) logged into the SAAS portal or website - it redirect the users to customization where they found the default and self-customization on each customization section (according to section 2- customization rules). In other words, all of the service cutomization is done by vendors and users(i.e. default and customizable) inorder to provide or get the software services.

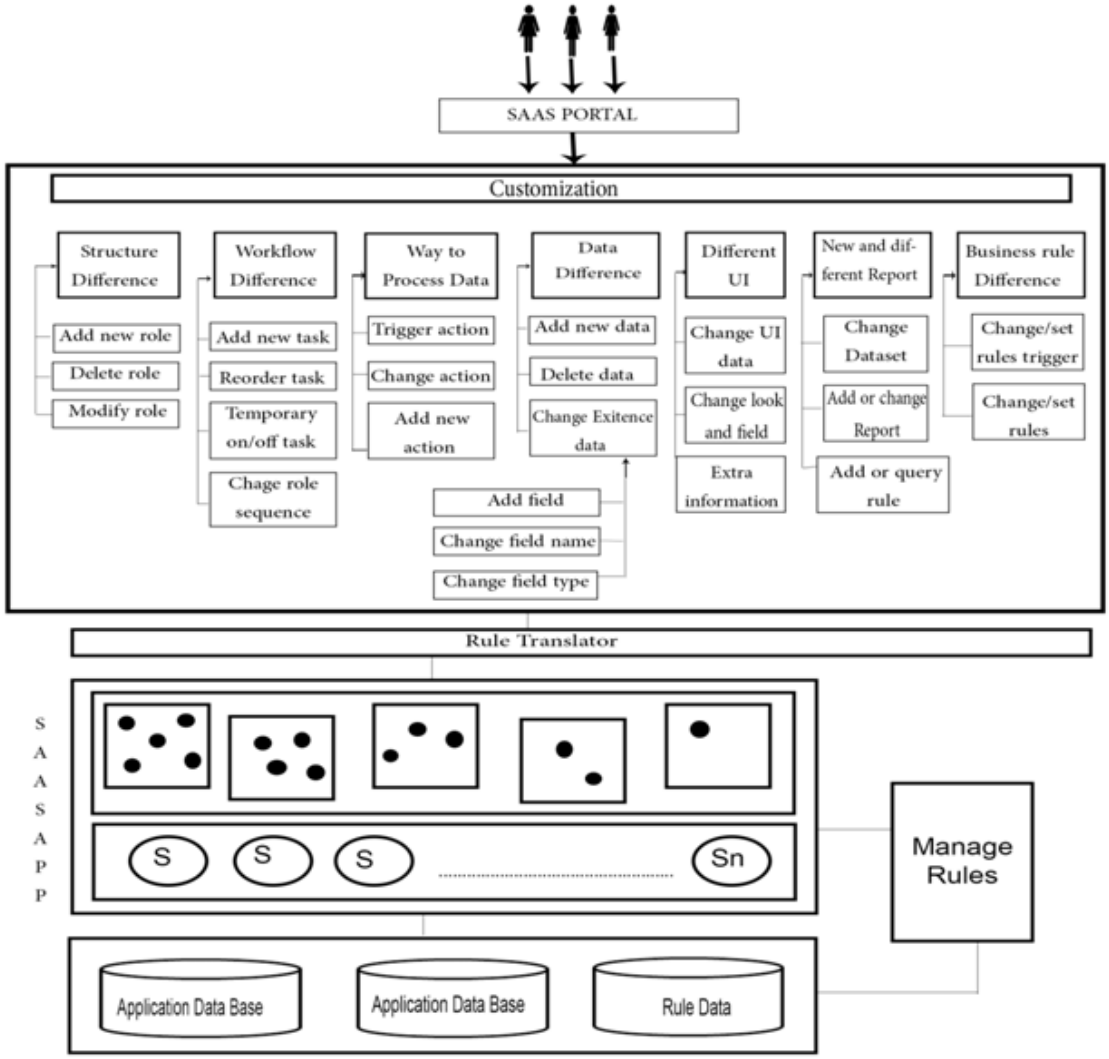

FIGURE 1.1: A formal framework for SAAS customization based on multi-layered architecture.

The customization selected customization process[11] is further translate and handle by the rule translators. Then, the SAAS apps[12] converted all the customization rules into processes and services which are shown as a user interface[10] to users, and store it on the database according to the boundary created by rule manager.

However, while adding new role to the all customization rules user follows the similar pattern and series of rule translator actions as shown in figure 1.2.

Role Name	Role description	Department	Staff

FIGURE 1.2 : Rule translator actions for customization (orgination structure difference).

Figure 1.2 shows the rule translator actions for customization which is completed by individual users for their customization processes according their organization structures.

Task Name	Process in List	Priority	Task State

FIGURE 1.3 : Rule translator actions for customization (workflow).

Figure 1.3 shows the rule translator actions for customization which is also completed by individual users for their customization processes according their workflow priority. Similarly, all the rules uses the same structure with differnent actions or only user required actions for the customization.

5. DEFINING CUSTOMIZATION VIA PROCESS ALGEBRA

*Algebraic Rules:*Here are the few algebraic rules used to define customizations and its processes and services [5, 6].

Process & services P = set of processes, P_g= General processes, P_t= Tenant Processes P_n=number of processes. S={s_1, s_2, s_3, s_4……S_n} Where, S= Set of Services, S_n = number of services
Constants a ε P for any atomic actions (a ε A). δ ε p deadlock (δ !ε A) ε P Silent Action(!ε A) (Here A is set of atomic actions.)
Functions + : Alternative Composition (sum). . : Sequential Composition (product). || : Parallel Process
Axiomatic rules X + y = y + X (X + y) + z = x + (y + z) X + X = X (X + y) z = xz +yz (xy)z =x(yz) x+δ = x δx=δ

TABLE 1: Algebraic Rules.

6. CUSTOMIZATION VIA PROCESS ALGEBRA

Here we will apply process algebra [5] on the customization of workflow process of tenant service processes. As an important part of tenant process, we will show the workflow process calculi customization approach in this paper. And all other processes are defined similarly and followed by the same customization method.

Here is the customization process of tenant work flow process according to algebraic rule (from Table 1). Here, P_t = {P_{t1}, P_{t2}, P_{t3}, P_{t4}, P_{t5}, P_{t6}, P_{t7}}, Where P_t is a set of processes for a tenant, and P_{t1}, P_{t2}, P_{t3}, P_{t4}, P_{t5}, P_{t6}, P_{t7} are organization structure, workflow, process data difference, data difference, user interface, report difference and business role difference simultaneously.

Similarly, $P_{t2} = \{P_a, P_b, P_c, P_d\}$, Where P_{t2} is set of processes for workflow. Here P_a, P_b, are reorder task and add task simultaneously.

$P_a = a \,.\, b\ (c + d)$……(i)
Where a,b,c,d are atomic actions.

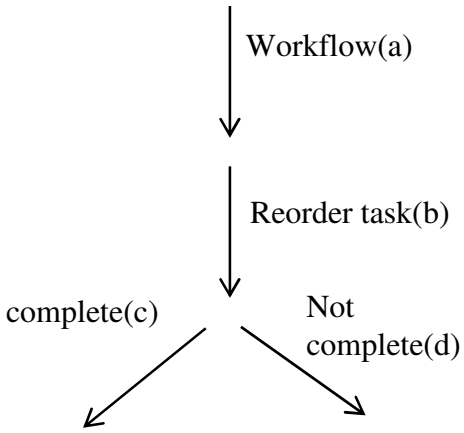

FIGURE 2.1: Reorder task –(i)

$P_b = a \,.\, b\ (c + d)$……(ii)
Where a,b,c,d are atomic actions

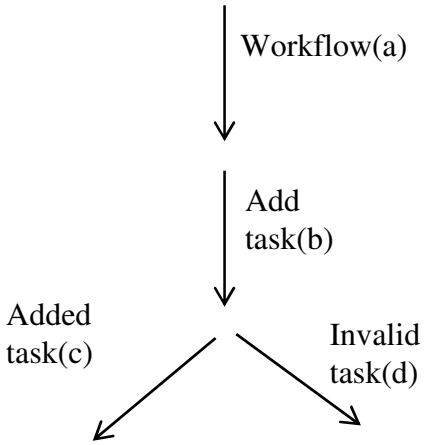

FIGURE 2.2: Add task – (ii)

And P_c, P_d are role sequence and on/off task simultaneously

$P_c = a \,.\, b\ (c + d)$ ……(iii)
Where a,b,c,d are atomic actions(i.e. workflow).

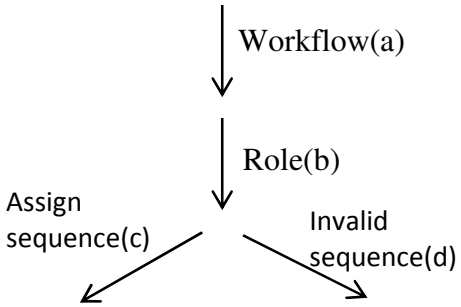

FIGURE 2.3: Role Sequence – (iii)

$P_b = a \,.\, b\ (c.e + d.f)$……(iv)
Where a,b,c,d and f are atomic actions

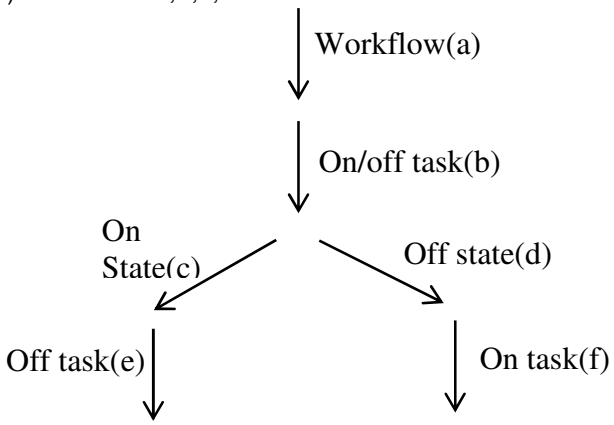

FIGURE 2.4: on/ off task – (iv)

Now we are going to present the SAAS architecture processes in the form of process algebra, here is the algebraic representation of the rule translator actions for customizing the add new role for tenant (orginazation structure difference - from figure 1.1) is:

$a.b(C_1(S_1+S_2+\ldots+S_n) + C_2(S_1+S_2+\ldots+S_n) + \ldots\ldots\ldots\ldots\ldots+C_n(S_1+S_2+\ldots+S_n))$……(v) Where,

a is role name, b is role description, C set of department i.e($C_1, C_2 \ldots C_n$)
and S is set of staffs i.e($S_1, S_2 \ldots S_n$)

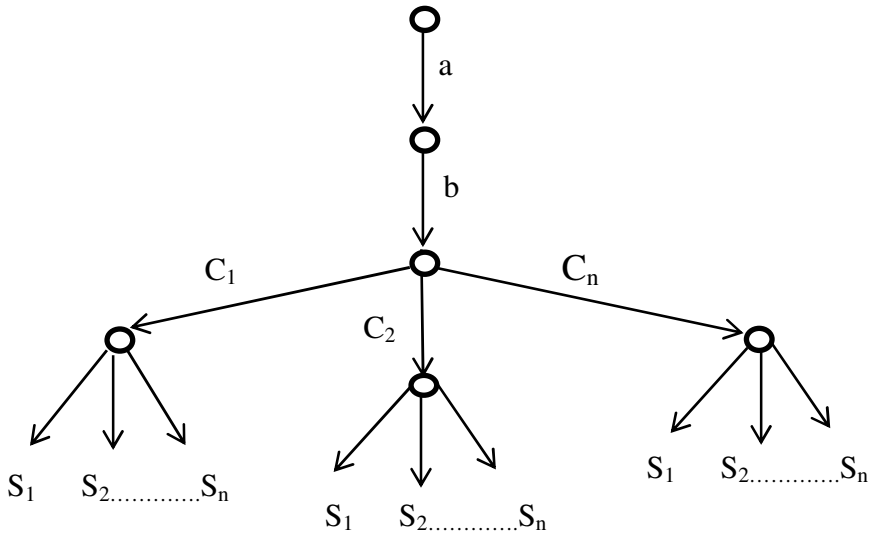

FIGURE 2.5: Algebraic Representation of Add New Roles – (v).

Similarly, the algebraic representation of figure 1.3 (workflow) based on the different priority.

Here: a.b.c ……(vi), a(bd+ce) ……(vii), a(de+cg+bg) ……(viii)
and (a(bd+ce)+je)f ……(ix) where a, b, c, d, e, f are different processes.

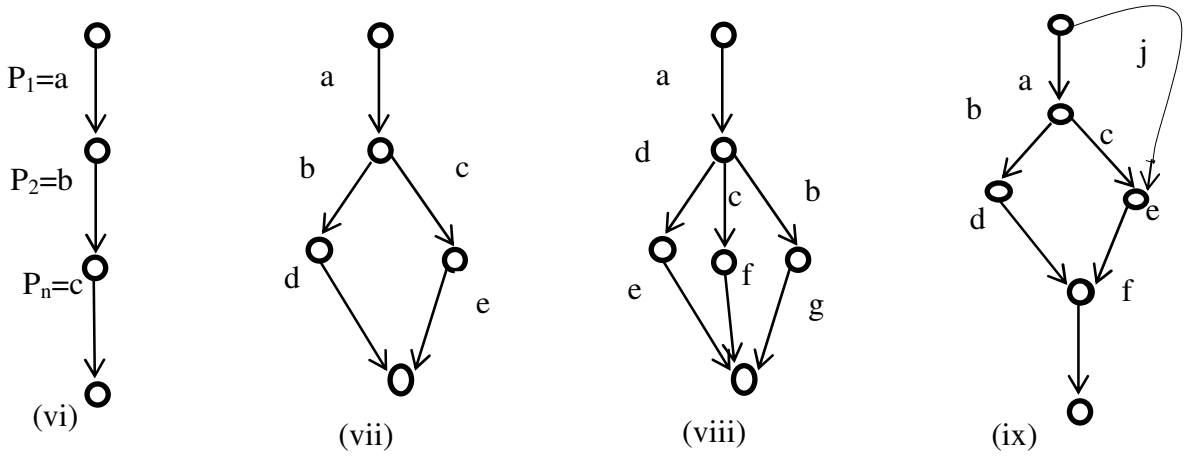

FIGURE 2.6: Algebraic Representation of Workflow.

7. CONCLUSION AND FUTURE WORK

Overall, we present the customization of the software as a services architecture as both vender and tenant customization format as well as the method for transforming the customization into the process algebra. Furthermore, we define and transform the organization structure and workflow of tenant process which can be further implemented and applied into all other tenant's or vendors processes via process algebra.

The architecture is presented for the first time in this paper, which is quite simple and easily adaptable for all kinds of software service providers. It's further implemented in the form of process algebra as well as SAAS multil-layered architecture for both users and vendors.

8. REFERENCES

[1] W. sun, X. Zhang, C. J. Guo, P. Sun, H. Su, Software-as-a-services: configuration and Customization perspectives, IEEE congress on services part-II, 2008 IBM China Research Lab, Beijing.

[2] W. Chen, B. Shen, Z. Qi, Template-based Business Logic Customization for SaaS Applications, Shanghai Jiao Tong University.

[3] T. v. Eijndhoven, M. E. Iacob, M. L. Ponisio, "Achiveing business process flexibility with business rules" 12th International IEEE Enterprise Distribute Object Computing Conference 2008.

[4] X. zhu, S Wang, Software Customization Based On Model-driven Architecture over SAAS Platforms.

[5] Z. Haomin, Y Guisheng, S. Changsong, Process Algebra Based for requirements process reorganization, International conference on Computer Science and Software Engineering, 2008.

[6] J.A. Bergstra, JW klop, Process Algebra: Specification and verification in bisimulation semantics.

[7] D. Chen, Q. Li, L. Kong, Process customization framework in SaaS Applications, 10th information system and application conference, 2013.

[8] K. Zhang, Y. shi, Q. Li, J. Bian, Data Privacy Preserving Mechanism based on Tenant customization for SaaS, International conference on multimedia information networking and security, 2009.

[9] W. T. Tsai, Q. Shao, W. Li, Ontology-based Intellegent customization framework for SaaS, IEEE conference on service oriented computing and applications, Perth, Australia, 2010.

[10] Nitu: Configurability in SaaS(Software as Service) applications. In 2nd annual conference on India Software Engineering Conference(ISEC '09). Pp. 19-26 (2009).

[11] H. Li, Y. Shi, and Q. Li. A multi-granularity customization relationship model for saas, 2009. IEEE Computer Society.

[12] S. J. Kang, SW Kang, SJ Hur, A Design of the Conceptual Architecture for a Multitenant SaaS Application Platform, DOI: 10.1109/CNSI.2011.56 · Source: IEEE Xplore, 2011.